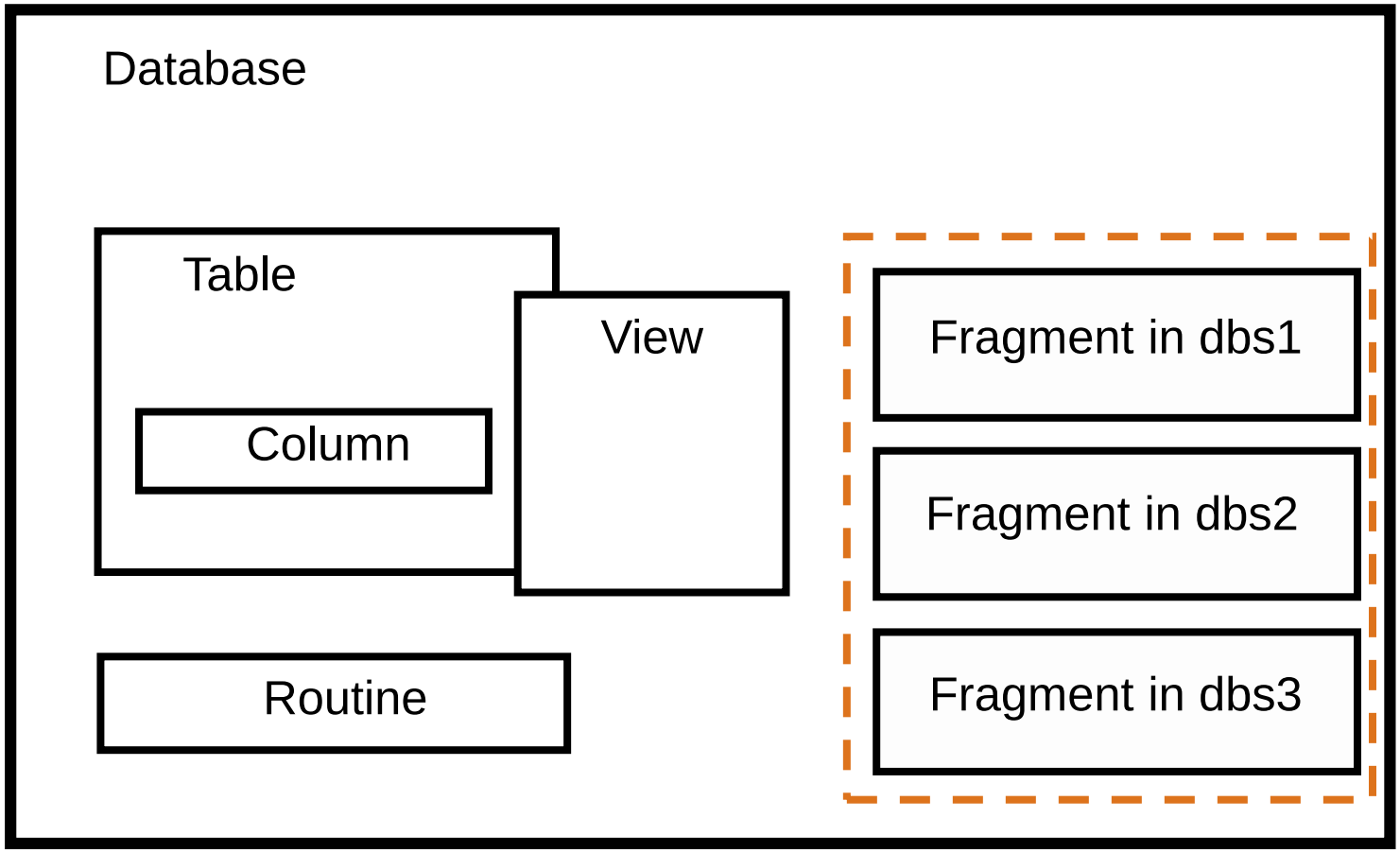
Data security

Unit objectives

- Use the database, table, and column-level privileges
- Use the GRANT and REVOKE statements
- Use role-based authorization

Levels of data security



Database-level privileges

- The three levels of database access are:
  - Connect
  - Resource
  - DBA

Table and column-level privileges

ALTER	Add, delete, or modify columns
DELETE	Remove rows from a table
INDEX	Create indexes for a table
SELECT	Retrieve information from the columns in a table
UPDATE	Modify information in the columns of a table
INSERT	Insert rows into a table
REFERENCES	Reference columns in referential constraints
ALL	Perform any or all of the preceding operations

Default privileges

- Database level:
  - When you create a database, you are automatically granted DBA privileges
- Table level:
  - Non-ANSI databases:
    - All table-level privileges except ALTER and REFERENCES granted to all users
    - Can use environment variable NODEFDAC to grant no privileges
  - MODE ANSI databases:
    - No default privileges granted

Granting database-level privileges

- Examples:

GRANT CONNECT TO PUBLIC;

CONNECT is granted to all users.

GRANT RESOURCE TO maria, joe;

GRANT DBA TO janet;

Revoking database-level privileges

- Examples:

```
REVOKE CONNECT FROM mike;

REVOKE RESOURCE FROM maria;
```

Granting table-level privileges

- Examples:

```
GRANT ALL ON customer TO PUBLIC;

GRANT UPDATE ON orders TO liz
  WITH GRANT OPTION;

GRANT INSERT, DELETE ON items TO mike
  AS maria;
```

Allows liz to grant update to other users

Grantor becomes maria

Revoking table-level privileges

- Examples:

```
REVOKE ALL ON orders FROM PUBLIC;

REVOKE DELETE, UPDATE ON customer
  FROM mike, maria;

REVOKE INSERT, UPDATE ON items
  FROM mike AS maria;
```

Revoker becomes maria

Granting column-level privileges

- Only SELECT, UPDATE, and REFERENCES privileges can be granted to individual columns.
- Column-level privileges are granted in the same way that table-level privileges are granted, except that a column list must follow the privilege in the GRANT statement.
- Examples:

```
GRANT SELECT (company, fname, lname)
 ON customer TO PUBLIC;
GRANT INSERT, UPDATE (quantity), SELECT
  ON items TO maria;
```

Routine privileges

- Examples:

```
GRANT EXECUTE ON total_orders TO PUBLIC;

GRANT EXECUTE ON square ( x INT ) TO maria;

REVOKE EXECUTE ON cancel_orders FROM joe, tom;
```

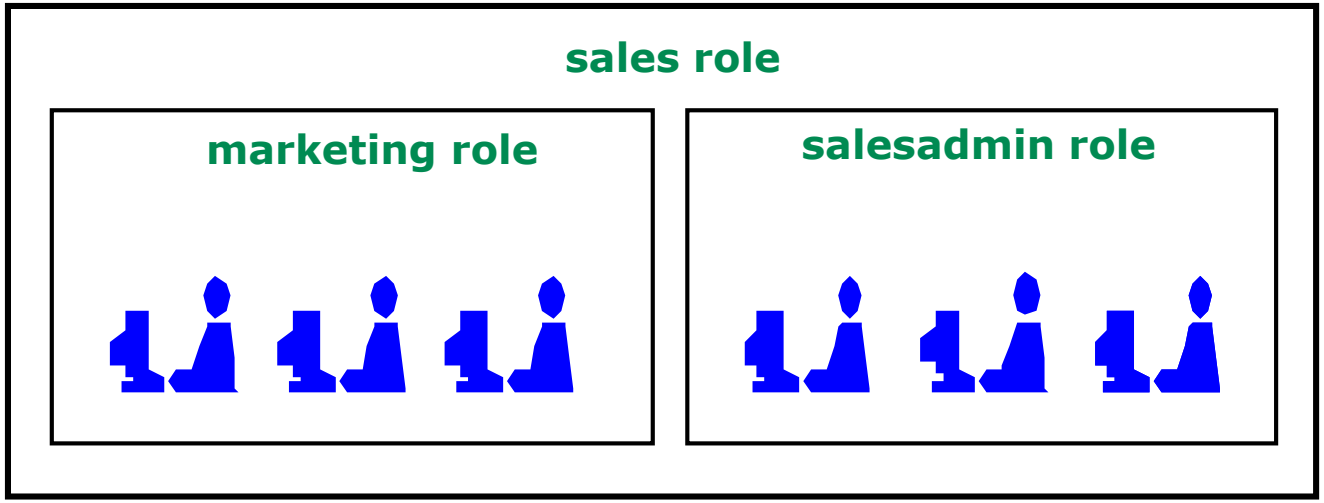
DataBlade privileges

- Examples:

```
GRANT EXTEND TO maria;

REVOKE EXTEND FROM joe, ram;
```

Roles



Creating roles

- Examples:

```
CREATE ROLE mkting;
CREATE ROLE slsadmin;
CREATE ROLE sales;

GRANT mkting TO jim, mary, ram;
GRANT slsadmin TO andy, liz, sam;
GRANT sales TO mkting, slsadmin;
```

Using roles (1 of 2)

- Examples:

```
REVOKE ALL ON orders FROM public;

GRANT SELECT ON orders TO sales;

GRANT INSERT, UPDATE, DELETE ON orders
  TO slsadmin;
```

What permission does the marketing role have on the orders table?

Using roles (2 of 2)

- A user can either inherit a default role, or specify a role to use in their session:
  - Default roles are assigned by the DBA using the GRANT ROLE statement:

    GRANT DEFAULT ROLE slsadmin TO liz;
  - A user can set their own role through the SET ROLE SQL statement:

    SET ROLE slsadmin;

    SET ROLE DEFAULT;
- Default roles can be granted to PUBLIC:

  GRANT DEFAULT ROLE slsadmin to PUBLIC;
- Default roles can be revoked with the REVOKE statement:

  REVOKE DEFAULT ROLE FROM ram;

GRANT and REVOKE FRAGMENT

- Examples:

```
REVOKE ALL ON orders
  FROM PUBLIC;
GRANT SELECT ON orders
  TO PUBLIC;

REVOKE FRAGMENT ALL
  ON orders
  FROM user1;

GRANT FRAGMENT INSERT,
    UPDATE,DELETE
  ON orders(dbspace1)
  TO user1;
```

All fragments of orders are read-only by user1.

user1 can now INSERT, UPDATE, or DELETE from only the fragment in dbspace1.

Discussion

- The orders table is fragmented so that orders for customer numbers 1 - 10,000 are in dbspace1 and orders for customer numbers 10,001 - 20,000 are in dbspace2.
- Given the GRANT and REVOKE FRAGMENT statements on the previous page, which of these statements would fail (if executed by user1)?

```
INSERT INTO orders(cust_nbr) VALUES 100;

SELECT * FROM orders;

UPDATE orders SET cust_nbr = 12200;

WHERE cust_nbr = 220;
```

Exercise

Data security

- assign and revoke privileges at the user and role levels

Unit summary

- Use the database, table, and column-level privileges
- Use the GRANT and REVOKE statements
- Use role-based authorization